CLAIMS

What is claimed is:

1. An emulation and native language interface test method comprising:

initializing an emulation language virtual machine;

wrapping native language code in a simulation test macro which creates simulated interfacing problems; and

examining reaction to said simulated interfacing problems when an emulation language application is run.

2. An emulation and native language interface test method of Claim 1 wherein said emulation language virtual machine creates a runtime environment and said runtime environment can include a class loader subsystem and an execution engine.

3. An emulation and native language interface test method of Claim 1 wherein said simulated test problems include simulations of error conditions associated with a native language code method attempt to respond to a call from emulation language code.

4. An emulation and native language interface test method of Claim 1 further comprising forwarding an indication that there is a insufficient memory allocation exception to a native language method attempting to ascertain an indication of a memory location for information associated with a native language function.

5. A Java native interface testing system comprising:

means for communicating information;

means for processing said information, including instructions for testing a Java native interface, said means for processing said information coupled to said means for communicating information; and

means for storing said information, including said instructions for testing said Java native interface, said means for storing said information coupled to said means for communicating information.

6. The Java native interface system of claim 5 wherein said means for processing performs a Java Native Language Interface test method.

7. The Java native interface system of claim 5 wherein said instructions include an interface testing macro module.

8. The Java native interface system of claim 5 emulates a Java virtual machine.

9. A Java Native Language Interface test method comprising:

investigating Java Native Language Interface test mode status;

running a Java application with simulated Java Native Language Interface problems if said Java Native Language Interface test mode is enabled; and

initiating a call to a Java Native Language Interface function directly without said simulated Java Native Language Interface problems if said Java Native Language Interface test mode is not enabled.

10. A Java Native Language Interface test method of Claim 9 wherein said Java Native Language Interface test mode status indicator indicates if said Java Native Language Interface test mode status is enabled.

11. A Java Native Language Interface test method of Claim 9 wherein said Java Native Language Interface test mode status indicator is a flag wherein a state of said flag indicates if said Java Native Language Interface test mode status is set.

12. A Java Native Language Interface test method of Claim 9 wherein a register value indicates said Java Native Language Interface test mode status.

13. A Java Native Language Interface test method of Claim 9 wherein said problems include identifying indications of Java Native Language Interface code trouble associated with out of memory situations.

14. A Java Native Language Interface test method of Claim 9 wherein a Java Native Language Interface problem simulation process is performed to simulate Java Native Language Interface problems.

15. A Java Native Language Interface test method of Claim 9 further comprising:

determining a Java Native Language Interface problem simulation occurrence level;

introducing simulation randomness;

performing an analysis whether to initiate a simulation of Java Native Language Interface problem;

calling a Java Native Language Interface memory allocation function normally;

forwarding a Java Native Language Interface problem indicator automatically; and

implementing a reaction to the Java Native Language Interface problem indication.

16. A Java Native Language Interface test method of Claim 15 further comprising:

looking up a predefined Java Native Language Interface problem simulation occurrence level;

generating a random value; and

correlating said random value to said JNI problem simulation occurrence level.

17. A Java Native Language Interface test method of Claim 16 further comprising:

comparing said randomly generated value to said Java Native Language Interface problem simulation occurrence level; and

initiating a simulation of a Java Native Language Interface problem if said generated value from is less than said Java Native Language Interface problem simulation occurrence level.

18. A Java Native Language Interface test method of Claim 16 further comprising initiating a controlled shut down.

19. A Java Native Language Interface test method of Claim 16 further comprising clearing a system and canceling information inventory collections that is occupying memory space.

20. A Java Native Language Interface test method of Claim 16 further comprising providing an indication of the Java Native Language Interface problem to a user.